Buch: Softwareentwicklung für die Naturwissenschaften

Dr. habil. Till Biskup

— Glossar zu Kapitel 01: „Motivation: Das Wesen der Wissenschaften“ —

Hinweis: Die nachfolgend genannten Begriffe und Definitionen erheben keinen Anspruch auf formale Korrektheit, sondern dienen lediglich dem besseren Verständnis der in der Vorlesung behandelten Themen und sind im jeweiligen Kontext zu sehen. Mehrfache, voneinander abweichende Definitionen in unterschiedlichen Kontexten sind daher möglich. Englische Begriffe werden zwar nach Möglichkeit übersetzt, erscheinen aber ggf. unter ihrem englischen Namen in der Liste. Verweise untereinander sind durch ↑ gekennzeichnet.

Anwenderdokumentation Art der Dokumentation, die sich in erster Linie an die Nutzer eines Programms wendet und deshalb deutlich weniger softwaretechnisch als eine ↑Entwicklerdokumentation formuliert ist, dafür aber meist von einer Kenntnis der für die Software relevanten Problemstellung ausgeht. Im Fokus steht im Gegensatz zur ↑Entwicklerdokumentation gerade *nicht* die Implementierung der Funktionalität, sondern ihre Nutzung. Typische Aspekte einer Anwenderdokumentation sind ein kurzer Überblick über die Merkmale eines Programms, einfache (einführende) Beispiele, konkrete Anwendungsszenarien und ggf. ein strukturiertes und umfangreiches Nutzerhandbuch. Eine gute Anwenderdokumentation ist mit erheblichem Aufwand verbunden, sorgt aber für eine deutlich bessere Nutzbarkeit (und damit Verbreitung) eines Programms.

API *application programming interface*, Programmierschnittstelle oder genauer ↑Schnittstelle zur Anwendungsprogrammierung

Clean Code „sauberer Code“, letztlich lesbarer Code, der insbesondere im Kontext der naturwissenschaftlichen Datenauswertung die essentiellen Kriterien von Wiederverwendbarkeit, Zuverlässigkeit und Überprüfbarkeit erfüllt.

Dokumentation im Kontext eines ↑Systems zur Datenverarbeitung mehrere Aspekte, angefangen von der ↑Anwenderdokumentation und ↑Entwicklerdokumentation verwendeter Software bis zum Protokoll aller Verarbeitungsschritte von Daten eines Datensatzes (↑Historie). Vgl. ↑selbstdokumentierend (1.).

Entwicklerdokumentation Art der Dokumentation, die sich in erster Linie an Entwickler, d.h. Programmierer, wendet und auf die Weiterentwicklung der Software fokussiert. Wesentlicher Bestandteil einer solchen Dokumentation ist i.d.R. die ↑API-Dokumentation, die sich meist automatisch aus den Kommentarköpfen im Quellcode generieren lässt. Die Aspekte einer externen Projektdokumentationund Ideen für die weitere Planung sind oft Teil einer solchen Entwicklerdokumentation, sowie die Beschreibung derjenigen Alternativen, die betrachtet aber nicht gewählt wurden. Bei einer Bibliothek ist die Entwicklerdokumentation relevant für die Nutzung derselben zur Entwicklung darauf basierender eigener Programme. Vgl. ↑Anwenderdokumentation.

größeres Projekt hier: Alles, was mehr als zwei Wochen Arbeit kostet und deutlich mehr als zweihundert Zeilen (reinen) Quellcode bzw. mehr als eine Handvoll Unterfunktionen umfasst. Wichtig ist der Fokus: Sobald ein Programm über längere Zeit und/oder von anderen verwendet werden soll (was eher die Regel statt die Ausnahme ist), ist es ein größeres Projekt.

Historie im Kontext eines ↑Systems zur Daten-

verarbeitung ein (vollständiges) Protokoll aller Prozessierungsschritte auf den Daten eines Datensatzes. Das Protokoll sollte neben *allen* Parametern eines Prozessierungsschrittes auch die Versionsnummer der jeweils verwendeten Routine enthalten, die in Verbindung mit einer ↑Versionsverwaltung erlaubt, genau die jeweils verwendete Fassung wiederherzustellen bzw. zu untersuchen. Notwendige Voraussetzung für ↑reproduzierbare Wissenschaft.

Infrastruktur personelle, sachliche und finanzielle Ausstattung, um ein angestrebtes Ziel zu erreichen. Im Kontext der Softwareentwicklung die Gesamtheit der Hilfsmittel, die (manche) Abläufe formalisieren und für Struktur und Überprüfbarkeit sorgen. Erleichtert die Arbeit des Programmierers, indem sie viele Aspekte festlegt, die so zur Routine werden (und keine Denkleistung absorbieren).

Kristallkugel Nur in der Theorie funktionierendes Hilfsmittel für den Blick in die Zukunft, das u.a. hilfreich wäre, um Software bereits in ihrer Entstehung auf künftige Anforderungen hin auszulegen. Aufgrund anderer damit einhergehender Probleme ist die reale Funktionalität einer K. nicht wünschenswert.

Lizenz Nutzungsrecht; Software ist *per se* vom Urheberrecht geschützt, unabhängig von ihrer Funktionalität. Lizenzen übertragen Nutzungsrechte vom Urheber der Software an ihren Nutzer.

Modularisierung Aufteilung der Gesamtaufgabe in kleinere Abschnitte. Die Aufteilung wird so lange fortgesetzt, bis die Lösung für den aktuellen Abschnitt unmittelbar in Form von Quellcode offensichtlich ist. Setzt die Definition von ↑Schnittstellen voraus.

monolithisch aus einem Stück bestehend; zusammenhängend und fugenlos

Nebenwirkung *side effect*, Wirkung; in der theoretischen Informatik die Veränderung des Programmzustands einer abstrakten Maschine. In der Praxis der Programmierung meist die Auswirkung einer Zuweisung eines Wertes zu einer Variablen, die außerhalb des konkret betrachteten Kontextes liegt.

numerische Genauigkeit Zahlen lassen sich im Allgemeinen in einem Rechner nicht mit beliebiger Genauigkeit repräsentieren. Diese eingeschränkte numerische Genauigkeit kann teilweise zu erheblichen Problemen (und fehlerhaften Ergebnissen) führen.

Nutzerdokumentation Eine aktuelle, auf die Bedürfnisse des Anwenders zugeschnittene (ggf. kurzgefasste) Dokumentation, nach Möglichkeit mit realen Anwendungsbeispielen. Hilft gerade bei umfangreicheren Auswertungsprogrammen und bei Nutzern, die keinen direkten Zugang zum Entwickler haben, die Software aber trotzdem verwenden wollen.

Physikalische Chemie Wesentliche Charakteristika sind die Ordnung des experimentell gewonnenen Erfahrungsmaterials mithilfe der theoretischen, numerischen und experimentellen Methoden der Physik und der Theoretischen Chemie, das Auffinden qualitativer Zusammenhänge und das Aufstellen quantitativer Beziehungen. Kurz gefasst: In der Physikalischen Chemie geht es um das Verständnis der Grundlagen und Hintergründe. Entsprechend stehen Daten und ihre Auswertung ($\Rightarrow$ Verständnis) (oft) im Zentrum. Datenauswertung erfolgt heute meist rechnergestützt, entsprechende Kenntnisse im Programmieren und dem Umgang mit komplexer(er) Software sind deshalb zunehmend wichtig.

Regression erneutes Auftreten eines einmal behobenen Fehlers in einer neuen Version. Lässt sich grundsätzlich dadurch umgehen, dass einmal detektierte Fehler in ↑Tests, meist ↑Unittests, verwandelt werden und jede zukünftige Version einer Software entsprechend (automatisch) auf das Auftreten dieses Fehlers überprüft wird.

Replizierbarkeit *replicability*, unabhängige Wiederholung der (Roh-)Datenerhebung, meist in Form von Experimenten und Beobachtungen, entsprechend nicht in jedem Fall durchführbar. Vgl. ↑Reproduzierbarkeit.

reproduzierbare Wissenschaft *reproducible science*, seit der Etablierung rechnergestützter Datenauswertung eigentlich nie mehr erreichter, aber für die Wissenschaft konstituierender Aspekt, dass sich Ergebnisse und Auswertungen unabhängig reproduzieren lassen, weil alle dazu notwendigen Aspekte vollständig und ausreichend beschrieben wurden. Motivation für die Vorlesung, deren Ziel es ist, die Hörer mit Konzepten vertraut zu machen, die letztlich eine ernstzunehmende reproduzierbare Wissenschaft ermöglichen.

Reproduzierbarkeit *reproducibility*, vollständige Wiederholbarkeit einer beschriebenen Datenverarbeitung und -Analyse. Ausgangspunkt sind existierende Daten, entsprechend sollte sie in jedem Fall möglich sein. Vgl. ↑Replizierbarkeit.

Robustheit hier: Toleranz einer Software gegenüber fehlerhaften Nutzereingaben. Insbesondere sollte die Software in einem solchen Fall nicht einfach abstürzen und damit die bisherige Arbeit des Nutzers vernichten. Siehe auch ↑Validierung.

Routine 1. eingeschliffener, ggf. strukturierter Arbeitsablauf; 2. (Software) Funktion: der Programmiersprache unter einem festen Namen bekannte Liste von Anweisungen, die eine bestimmte Aufgabe erfüllt.

Schnittstelle mehrfache, leicht unterschiedliche Bedeutungen im Kontext der Softwareentwicklung; hier: Verbindung zwischen einem Stück Software (Programm, Routine) und seiner Umgebung. Dient der Trennung von Verantwortlichkeiten und ermöglicht ↑Modularisierung.

selbstdokumentierend hier: Eigenschaft von Auswertungssoftware, alle Prozessierungsschritte (automatisch) zu dokumentieren. Wesentliche Voraussetzung für die Nachvollziehbarkeit wissenschaftlicher Datenverarbeitung und -Auswertung.

Softwarearchitektur Aufteilung eines größeren Projektes in einzelne kleinere Projekte bzw. Aufgaben (↑Modularisierung), Definition klarer ↑Schnittstellen und Anforderungen sowie der Interaktion der einzelnen Teile miteinander. Nach Robert C. Martin die Gestalt eines Systems, die ihm von seinen Entwicklern gegeben wird: Unterteilung des Systems in Komponenten, ihre Anordnung, und die Art ihrer Interaktion miteinander. [1, S. 136]

System zur Datenverarbeitung hier: Gesamtsystem für wissenschaftliche Datenverarbeitung von der Datenaufnahme bis zur fertigen Publikation, das alle Aspekte umfasst und das ↑reproduzierbare Wissenschaft möglich macht und gewährleistet. Definitiv ein ↑größeres Projekt, das nicht nur eine ↑monolithische Anwendung umfasst, sondern viele Aspekte darüber hinaus. Setzt entsprechende ↑Infrastruktur und in der Umsetzung der einzelnen Komponenten sauberen Code (↑Clean Code) und eine solide ↑Softwarearchitektur voraus.

Test hier: strukturiertes Vorgehen, eine Software zu überprüfen. Setzt die Definition klarer Anfangs- und Endbedingungen (Eingabe und Ergebnis) voraus und sollte idealerweise vollständig automatisiert ablaufen können.

Unittest ↑Test eines Codeblocks in Isolation. Ein Unittest überprüft von außen, ohne den Quellcode des zu testenden Systems zu kennen oder zu benötigen. Die getesteten Codeblöcke sind i.d.R. klein. Zwingende Voraussetzung ist, dass das (erwünschte) Verhalten des zu testenden Codeblocks eindeutig definierbar (und seinerseits in Form von Quellcode formalisierbar) ist. Unittests sind gewissermaßen die unterste Ebene (automatisierter) ↑Tests.

Validierung hier: Überprüfung der Nutzereingaben eines Programms auf Sinnhaftigkeit und gültigen Wertebereich. Wesentlicher Aspekt der ↑Robustheit von Software.

Versionsnummer hier: eindeutige Bezeichnung einer Version einer Software, deren Kenntnis es erlaubt, auf genau diese Version der Software Bezug zu nehmen.

Versionsverwaltung Software zur Verwaltung unterschiedlicher Versionen von Dateien und

Programmen, die den Zugriff auf beliebige ältere als Versionen gespeicherte Zustände ermöglicht. Gleichzeitig ein wichtiges Werkzeug für die Softwareentwicklung und wesentlicher Aspekt einer Projektinfrastruktur.

Vorlagen *templates*, hier: Von der jeweiligen Software zur Datenverarbeitung weitgehend unabhängige Dokumente, die der strukturierten Präsentation von Daten bzw. Ergebnissen dienen. In gewisser Weise ein Lückentext, der automatisch ausgefüllt werden kann. Zentraler Aspekt ist die Trennung der Formatierung von der Software, die die Platzhalter ersetzt.

Wissenschaft Auf den Erkenntnisgewinn ausgerichtetes, systematisches menschliches Unterfangen, das in der Regel eine Reihe von Kriterien erfüllt bzw. erfüllen sollte: Unabhängigkeit vom Beobachtenden bzw. Durchführenden, gegründet auf den Erkenntnissen früherer Generationen, sowie überprüfbar, nachvollziehbar und ggf. reproduzierbar.

Literatur

[1] Robert C. Martin. *Clean Architecture. A Craftman's Guide to Software Structure and Design*. Boston: Prentice Hall, 2018.